AFRL-VA-WP-TP-2003-306

TASK ALLOCATION FOR WIDE AREA SEARCH MUNITIONS WITH VARIABLE PATH LENGTH

Corey Schumacher
Phillip R. Chandler
Steven J. Rasmussen
David Walker

MARCH 2003

AIR VEHICLES DIRECTORATE
AIR FORCE RESEARCH LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7542

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YY)* March 2003	**2. REPORT TYPE** Conference Paper Preprint	**3. DATES COVERED** *(From - To)*

4. TITLE AND SUBTITLE TASK ALLOCATION FOR WIDE AREA SEARCH MUNITIONS WITH VARIABLE PATH LENGTH	**5a. CONTRACT NUMBER** In-house
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER N/A
6. AUTHOR(S) Corey Schumacher and Phillip R. Chandler (AFRL/VACA) Steven J. Rasmussen (Veridian, Inc.) David Walker (Brigham Young)	**5d. PROJECT NUMBER** N/A
	5e. TASK NUMBER N/A
	5f. WORK UNIT NUMBER N/A
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Control Theory Optimization Branch (AFRL/VACA) Control Sciences Division Air Vehicles Directorate Air Force Research Laboratory, Air Force Materiel Command Wright-Patterson Air Force Base, OH 45433-7542 — Veridian, Inc. — Brigham Young University	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFRL-VA-WP-TP-2003-306
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Vehicles Directorate Air Force Research Laboratory Air Force Materiel Command Wright-Patterson Air Force Base, OH 45433-7542	**10. SPONSORING/MONITORING AGENCY ACRONYM(S)** AFRL/VACA
	11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S) AFRL-VA-WP-TP-2003-306

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
To be presented at the American Control Conference, Denver, CO, June 4 - 6, 2003.
© 2003 AIAA. This work is copyrighted. The United States has for itself and others acting on its behalf an unlimited, paid-up, nonexclusive, irrevocable worldwide license. Any other form of use is subject to copyright restrictions.

14. ABSTRACT
This paper addresses the problem of task allocation for wide area search munitions. The munitions are required to search for, classify, attack, and verify the destruction of potential targets. It is assumed that target field information is communicated between all elements of the swarm. A network flow optimization model is used to develop a linear program for optimal resource allocation. This method can be used to generate a tour of several assignments to be performed consecutively, by running the assignment iteratively and only updating the assigned task with the shortest estimated time of arrival (ETA) in each iteration. Periodically re-solving the overall optimization problem as new targets are discovered results in coordinated action by the search munitions. Variable path lengths are used to improve overall performance and prevent infeasibility. Simulation results are presented for a swarm of eight vehicles searching an area containing multiple potential targets.

15. SUBJECT TERMS
task allocation, cooperative control, wide area search munitions

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT:**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** (Monitor)
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	SAR	20	Phillip R. Chandler **19b. TELEPHONE NUMBER** *(Include Area Code)* (937) 255-8680

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18

TASK ALLOCATION FOR WIDE AREA SEARCH MUNITIONS WITH VARIABLE PATH LENGTH

Corey Schumacher, Phillip R. Chandler
Control Science Division
Air Force Research Laboratory (AFRL/VACA)
Wright-Patterson AFB, OH 45433-7531

Steven J. Rasmussen
Veridian Inc.
Wright-Patterson AFB, OH

David Walker
Mechanical Engineering Department
Brigham Young University
Provo, Utah 84602

Abstract

This paper addresses the problem of task allocation for wide area search munitions. The munitions are required to search for, classify, attack, and verify the destruction of potential targets. It is assumed that target field information is communicated between all elements of the swarm. A network flow optimization model is used to develop a linear program for optimal resource allocation. This method can be used to generate a "tour" of several assignments to be performed consecutively, by running the assignment iteratively and only updating the assigned task with the shortest estimated time-of-arrival (ETA) in each iteration. Periodically re-solving the overall optimization problem as new targets are discovered results in coordinated action by the search munitions. Variable path lengths are used to improve overall performance and prevent infeasibility. Simulation results are presented for a swarm of eight vehicles searching an area containing multiple potential targets.

Introduction

Autonomous wide area search munitions (WASM) are small, powered air vehicles, each with a turbojet engine and sufficient fuel to fly for a short period of time. They are deployed in groups, or "swarms," from larger aircraft flying at higher altitudes. They are individually capable of searching for, recognizing, and attacking targets. Cooperation between munitions has the potential to greatly improve their effectiveness in many situations. The ability to communicate target information to one another will greatly improve the capability of future search munitions.

In this paper we describe a time-phased network optimization model designed to perform task allocation for a group of powered munitions each time it is run. The model is run simultaneously on all munitions at discrete points in time, and assigns each vehicle one or more tasks each time it is run. The model is solved each time new information is brought into the system, typically because a new target has been discovered or an already-known target's status has been changed. The network optimization model is run iteratively so that all of the known targets will be completely serviced by the resulting allocation. Classification, attack, and battle damage assessment tasks can all be assigned to different vehicles when a target is found, resulting in the target being more quickly serviced. A single vehicle can also be given multiple task assignments to be performed in succession, if that is more efficient than having multiple vehicles perform the tasks individually.

A similar iterative network model for assigning multiple tasks was studied in [7], but that work has some limitations. A primary limitation of the work in [7] is that path planning is treated as if it were completely decoupled from task assignment. Due to the assumption of decoupling, there are cases where [7] fails to calculate feasible trajectories for task assignments when such feasible trajectories should in fact exist. As a result, some tasks are not assigned until the assignment algorithm is run again at a later time. This is inefficient, and delays the time when some tasks will be performed. In this paper, methods are presented to calculate minimum-length paths for any task assignment and any desired feasible arrival time. These variable-length path planning algorithms are combined with the iterative network flow task assignment algorithms presented in [7] to create a complete path-planning and task assignment algorithm that is guaranteed to compute feasible trajectories and assign all needed tasks to the available vehicles, assuming sufficient fuel remains to perform the tasks.

The cooperative control algorithm is implemented in a simulation with up to ten wide area search munitions and ten potential targets. This simulation has six degree-of-freedom dynamics for the search munitions and the capability to include a variety of target types. This paper presents simulation results for a swarm of vehicles searching an area containing a cluster of targets. The vehicles have limited flight times due to fuel constraints, and have an ATR capability. The vehicles are assumed to be able to communicate target state information to each other, as well as the calculated "benefits" for each vehicle performing each possible task.

Scenario

We begin with a set of N vehicles, deployed simultaneously, each with a life span of 30 minutes. We index them i = 1, 2, …, N. Targets that might be found by searching fall into known classes according to the value or "score" associated with destroying them. We index them with j as they are found, so that j = 1, 2, …M and Vj is the value of target j. We assume that there is no precise a priori information available about the number of targets and their locations. This information can only be obtained by the vehicles searching for and finding potential targets via Automatic Target Recognition (ATR) methodologies. The ATR process is modeled using a system that provides a probability that the target has been correctly classified. The probability of a successful classification is based on the viewing angle of the vehicle relative to the target. At this time, the possibility of incorrect identification is not modeled, but targets are not attacked unless a 90% probability of correct identification is achieved. Further details of the ATR methodology can be found in [3], and a detailed discussion is available in [4].

Network Optimization Model

Network optimization models are typically described in terms of supplies and demands for a commodity, nodes that model transfer points, and arcs that interconnect the nodes and along which flow can take place. To model weapon system allocation, we treat the individual vehicles as discrete supplies of single units, tasks being carried out as flows on arcs through the network, and ultimate disposition of the vehicles as demands. Thus, the flows are 0 or 1. We assume that each vehicle operates independently, and makes decisions when new information is received. These decisions are determined by the solution of the network optimization model. The receipt of new target information triggers the formulation and solving of a fresh optimization problem that reflects current conditions, thus achieving feedback action. At any point in time, the database onboard each vehicle contains a *target* set, consisting of indexes, types and locations for targets that have been classified above the probability threshold. There is also a *speculative* set, consisting of indexes, types and locations for potential targets that have been detected, but are classified below the probability threshold and thus require an additional look before striking. Figure 1 provides an illustration of this model.

The model is demand driven, with the large rectangular node on the right exerting a demand-pull of N units (labeled with a supply of –N), so that each of the munition nodes on the left (with supply of +1 unit each) must flow through the network to meet the demand. In the middle layer, the top M nodes represent all of the targets that have been identified with the required minimum classification probability at this point in time and thus are ready to be attacked. An arc exists from a specific vehicle node to a target node if and only if it is a feasible vehicle/target pair. At a minimum, the feasibility requirement would mean that there is enough fuel remaining to strike the target if tasked to do so. Other feasibility conditions could also enter in, if, for example, there were differences in the onboard weapons that precluded certain vehicle/target combinations, or if the available attack angles were unsuitable. The bottom R nodes of the middle layer represent all of the potential targets that have been identified, but do not meet the minimum classification probability. We call them *speculatives*. The minimum feasibility requirement for an arc to connect a vehicle /speculative pair is sufficient fuel for the vehicle unit to assume a position in which it can deploy its sensor to assist in elevating the classification probability beyond threshold. The lower tier models alternatives for battle damage assessment for targets that have been struck. Finally, each node in the vehicle set on the left has a direct arc to the far right node labeled sink, modeling the option of continuing to search. The capacities on the arcs from the target and speculative sets are fixed at 1. Due to the integrality property, the flow values are constrained to be either 0 or 1. Each unit of flow along an arc has a "benefit" which is an expected future value. The optimal solution maximizes total value.

The network optimization model can be expressed as:

$$\max J = \sum_{i,j} c_{ij} x_{ij} \tag{1}$$

Subject to:

$$\sum_{j,s} \left(x_{ij} + x_{is} \right) = 1, \qquad ,\forall i = 1,\ldots,n \tag{2}$$

$$x_{j,k} - \sum_{i} x_{i,j} = 0, \qquad \forall j = 1,\ldots,m \tag{3}$$

$$\sum_{i} x_{is} + \sum_{j} x_{jk} = n, \qquad ,n = \#UAVs \tag{4}$$

$$x \leq 1 \tag{5}$$

$$x \geq 0 \tag{6}$$

This particular model is a capacitated transshipment problem (CTP), a special case of a linear programming problem. Constraint (2) enforces the condition that each vehicle be assigned one task. Constraint (3) enforces the condition that flow-in must equal flow-out for all nodes. Constraint (4) forces the number of assigned tasks to be equal to the number of available vehicles. Constraints (5) and (6) enforce the binary nature of the problem. Any particular flow is either active or inactive (0 or 1). Restricting these capacities to a value of one on the arcs leading to the sink, along with the integrality property, induces binary values for the decision variables x_{ij}. Due to the special structure of the problem, there will always be an optimal solution that is all integer [2]. Solutions to this problem pose a small computational burden, making it feasible for implementation on the processors likely to be available on disposable wide area search munitions.

The goal of the optimization problem is to maximize the value of the tasks performed by the vehicles at the time the model is solved. Solving the model whenever new target information is available attempts to maximize the value of the targets destroyed over the life of the munitions.

Due to the integrality property, it is not normally possible to simultaneously assign multiple vehicles to a single target, or multiple targets to a single vehicle. However, using the network assignment iteratively, "tours" of multiple assignments can be determined. This is done by solving the initial assignment problem once, and only finalizing the assignment with the shortest ETA. The assignment problem can then be updated assuming that assignment is performed, updating target and vehicle states, and running the assignment again. This iteration can be repeated until all of the vehicles have been assigned terminal attack tasks, or until all of the target assignments have been fully distributed. The target assignments are complete when classification, attack, and battle damage assessment tasks have been assigned for all known targets. Assignments must be recomputed if a new target is found or a munition fails to complete an assigned task.

A potential complication arises from the decoupling between path planning and task assignment. Minimum-time trajectories are calculated for each vehicle to perform each needed task, and these are then sent to the assignment algorithm and used in calculating the task benefits c_{ij}. If the minimum-time trajectory does not satisfy the timing constraints imposed by previous tasks, a new path is calculated that will meet the timing constraints. This can occur for attack and verification tasks, but not classification, as classification is the first task that needs to be performed on a target.

Benefit Calculation

One of the critical questions involved in using the network flow model for coordinated control and decision-making for WASM is how the values of the benefits, or weights, *c(i,j)* are chosen. Different values will achieve good results for different situations. For example, reduced warhead effectiveness greatly increases the importance of battle damage assessment and potential repeated attacks on an individual target. A simplified scheme has been developed which does not attempt to address the full probabilistic computation of the various Expected Values. It is intended to assign the highest value possible to killing a target of the highest-valued type, with other tasks generating less of a benefit. Overall, the chosen weights tend to result in the least possible lost search time for the execution of a particular task. The values of different tasks are calculated as follows:

C(i,j) = Expected value of vehicle I attacking target j
= ((Probability target type has been correctly identified)*(Probability of destroying target j) * (Value of target j) – search value of time until attack)*memory weight
= $(P_{id}*P_k*V_j - \max(\text{target values})*T_a/T_m)*\gamma$

C(i,s) = Value of vehicle i continuing to search
= (Maximum Target Value)*(Remaining flight time)/(Maximum flight time)*memory weight
= $(\max(\text{target values})*T_f/T_m)*\gamma$

C(i,k) = Expected value of vehicle i assisting in classifying speculative k
= ((Probability successful ATR)*(Expected value of target being attacked after classification) + Value of continued search after classification) *(Previous task weighting))*memory weight
= $(Patr*Pk*Vj+ \max(\text{target values})*(T_f - T_{classify})/T_m)*\gamma$

C(i,g) = Expected value of vehicle i performing BDA on target g
= (((Probability successful BDA)*(Probability target was not killed)(Probability of correct target ID)(Value of target j) + Value of continued search after classification) *(Previous task weighting))*memory weight
= $((P_{bda}*(1-P_k)*P_{id}*V_j + \max(\text{target values}) *(T_f-T_{bda})/T_{mg}))*\gamma$

There are five possible target types with different values, and different ATR characteristics. P_{id} is an input based on the quality of the ATR recognition. T_f is the remaining available flight time of a vehicle, and T_m is the maximum flight time of the vehicle. For the following simulation results, some of the parameters were set as constants: $P_{id} = 0.90$, $P_k = 0.80$, $P_{bda} = 1.0$. $T_{classify}$ and T_{bda} are equal to the flight time to reach the specified target, plus the time needed to return to search after the task is completed.

The value of attacking a target is modified by the time required for a vehicle to perform that attack, so that a slightly higher value is associated with a vehicle that can perform the attack task sooner. The value of continuing to search is set such that the value of searching is equal to the value of killing a high-value target initially, and degrades linearly with search time remaining. This will tend to result in vehicles with less flight time remaining being used to kill targets, and vehicles with more fuel left being used to search, classify, and perform BDA. Determining precise appropriate values for the probabilities of successful ATR and BDA is difficult, and requires substantial modeling of those processes, which this paper does not address in substantial detail. Simplified models giving reasonable values for these parameters are used. The value of all possible tasks, vehicle, and target assignment combinations are calculated and sent to the capacitated transshipment problem solver. The values are multiplied by 1000 before being sent to the solver, as it only works with integers and rounding will result in poor results without the scaling factor.

The memory weighting γ is very important when recalculating old assignments after a new target is found, as will be illustrated in the simulation results. Without a memory weighting, small variations in calculated path lengths due to different initial conditions can result in assignments being changed when it would not in reality be efficient to do so. Normally, $\gamma = 1.0$. However, in successive assignment calculations, we use a small memory weighting ($\gamma = 1.05$) in calculating the benefit of a vehicle performing the particular non-search task which it is already performing (a classify, attack, or verification task on a specific target). This memory weighting greatly reduces the "churning" resulting from vehicles' assignments being changed unnecessarily.

Variable-Length Path Planning

Path planning is performed using kinematic geometry. Minimum length paths to any desired point and heading can be found using turn circles and straight line segments. We have developed algorithms for extending these minimum-length paths to achieve any desired feasible path length. The method of path elongation is different for each of five different cases. Which case must be used is determined by the initial relative position and heading of the vehicle with respect to the final destination.

Without loss of generality, the cases are defined with the initial vehicle velocity at a heading of zero degrees (going from left to right), with the target at the origin. For any given initial position and heading, with any arbitrary final target position, the coordinates are transformed to the zero initial heading with the target at the origin for classification within one of the five cases. The boundaries of the individual case types in the transformed coordinates are shown in Figure 4 on the following page. The case the vehicle is found to be in determines the method of elongation to be used and the window of all possible completion times for the given task.

The boundaries in the figure represent mathematical case limits for the vehicle relative to the goal, assuming the vehicle travels from left to right after the coordinate transformation. The limits on vehicle position for each case are primarily dictated by the position of the left and right turning circles for the vehicle. The distance from the target to these circle centers is needed in determining how the path can be elongated, and where path length discontinuities will occur during path elongation. The discontinuities arise when the inside turn circle center is sufficiently far from the goal at the vehicle's initial position, but then becomes too close to the goal during elongation; that is, when the inside turn circle is outside radial distance "b" initially (the distances represented are defined below), but crosses below this radial distance during the delay.

The thin green circles in the Figure 2 represent the limiting distances for the turning circle center distances, and are labeled from a to d. These radial distances from the target represent limitations on the vehicle such as minimum turning radius and required stand off distance for task completion. The radial distances shown are summarized as:

- "a" – Sensor Stand Off Distance, R; this is the distance the vehicle must be from the target in order for it's sensor to pass over the target.
- "b" – The minimum radial distance from target to final turn circle center for vehicle position to satisfy sensor stand off limit boundaries. This distance is given as $L = \sqrt{R^2 + r^2}$, for $r = TurnRadius$.
- "c" – This is a distance of $L + r$, and is used in defining Case *I* and the transitions and limits for Case *III*.
- "d" – A distance of $L + 2r$; This distance is a boundary for the outer turn circle, and determines whether the vehicle is in Case *III* or *IV*, and whether or not there will be discontinuities in the path length elongation.

Case Specific Elongation Methods

The defined cases and their associated methods are closely tied together. The elongation methods for the specific cases involved either adding straight segments to the path where the shortest path would turn, or turning immediately but in the opposite direction of the first turn for the shortest path, or a combination of an opposite turn and a straight segment.

Case Type *I*

A vehicle is in case *I* when it can elongate its path by any amount greater than zero by continuing straight on it's initial velocity heading. In this case the path length can be changed continuously, without any discrete jumps, to obtain any length between the shortest path length and infinity (with the only limit being the fuel of the vehicle). In Figure 5, the vehicle path shown on top is in case *I* and can elongate its path continuously. The vehicle path shown at the bottom of Figure 5 is too close to the target (distance perpendicular to velocity direction) to elongate continuously by continuing straight. An additional benefit obtained from continuing straight is that the path is elongated while also being able to continue searching for more targets along the vehicle's current path.

Case Type *II*

Case *II* is the only case in which a desired path length can be obtained directly. Figure 6 demonstrates how the path elongation is accomplished in Case *II*.The only condition for a vehicle to be in case *II* is that the vehicle must turn at least 180° in one direction. Whenever this is the case, a path extension equal to half the desired elongation distance can be added to the path on both sides of the 180° turn. This works even if the turn is completed through multiple waypoints. In Figure 6, the elongated paths are in solid lines and the original paths are shown in dashed lines. The case *II* elongation method has two attractive elongation characteristics due to the fact that the elongation occurs in the middle of the path. First, the final heading is unchanged, so subsequent tasks will not need to re-plan their routes based on the initial velocity direction for that task. The second beneficial attribute of the 180° turn path elongation is that, just like in Case *I*, the vehicle can create and follow paths of any length greater than or equal to that of the minimum path.

Case Type *III*:

The elongation method for Case *III* is illustrated in Figure 5. Case types *III* and *IV* are very similar. The only difference between them is that case *IV* is close enough to the target to cause discontinuities in the path length during elongation. This means that a vehicle in case *III* can find an acceptable path of any length longer than the minimum, but a vehicle in case *IV* will have a range of possible path lengths that it is not possible for the vehicle to obtain. The elongation method for cases *III* and *IV* is a two-part elongation involving both turning away and continuing straight (if the desired path length involves a large enough elongation of the shortest path). The length of the elongation will determine whether both methods are used, or if only the initial turning away. In both cases the vehicle requires only a single turn in the shortest length path. That is, the vehicle is far enough away from the target to turn directly towards it until it is facing the target. If the elongation involves both methods, a vehicle in case *III* will turn away until the vehicle's new position and heading fit under a case *I*. At this point the path elongation will switch to case *I* and iterate to find the elongated path by continuing straight. The point at which the vehicle transitions to case *I* is the critical point for the case *III* vehicle.

Case Type *IV*

The elongation method for Case *IV* is illustrated in Figure 6. Path elongation for a vehicle in case *IV* more complicated due to the path length discontinuities. The vehicle will perform in exactly the same manner as case *III* except when it is near/in the discontinuity. Due to the jump in path length there are two critical points in case *IV*, and two timing windows. The first critical point is where the discontinuity begins, and the path length associated with an elongation to this point is a bound on the upper value of the first timing window. If the desired path length is before the first critical point, then the algorithm iterates on an elongation between zero the first critical point to find the path. The second critical point is where the discontinuity ends, and the path associated with it is the lower bound on the second timing window. Once the vehicle enters too close to the target, the best course is to continue turning away until it reaches the second critical point. If the desired path length is in the discontinuity (and therefore infeasible), then the path through the second critical point is returned as the best feasible value. If the path is still not long enough after the elongation through the second critical point, the vehicle transitions to case *I* and iterates on a straight line elongation to find a path of the desired length.

Case Type *V*

The elongation method for Case *IV* is illustrated in Figure 7. Case *V* is basically a special instance of case *I*. When a vehicle is in case *V*, the shortest path to the goal always involves an immediate turn away from the goal. If the path was elongated in the same way as case *I* (by continuing search), the same algorithms do not apply without large modifications. These modifications require a different approach in the algorithms, making the code less general. The solution used to resolve this problem was to allow the vehicle to maintain the same course through the first turn in the opposite direction. Once the vehicle has completed this turn it is outside the complications, allowing case *I* functions to apply directly. At this point the vehicle begins to perform elongation by continuing straight. This method, like the nearly identical method for case *I*, produces a continuous range of possible path lengths. Again, the upper bound on the path length is constrained by the fuel limits of the vehicle. The resulting second turn in the path is always nearly, but not quite, 180°. This prevents the case from switching to case *II*, rather than case *I*. However, the large turn makes the iteration for case *I* work very quickly, requiring only two or three iterations to converge.

Iterative Path Elongation

Only one of the five cases results in a truly linear path elongation and can be solved directly. The other cases result in nonlinear elongations and direct solutions for a path of a specified length could not be found. To finding paths for these cases, it was resolved to use an iterative method much like a numerical Newton-Raphson search [8].

The functions that the Newton-Raphson-like method is working on depend on the vehicle's case. Different cases have paths that are more nonlinear than others. Also, the domain of the "function" varies since in some cases the elongation is obtained through flying straight, and in other cases it is obtained by turning in the opposite direction. The range of the functions is always the resulting path length (or can be measured in ETA since velocity and time are equivalent for the constant velocity vehicles). The domain is the amount the vehicle delays its approach to the target through the elongation method. In case *I*, where elongation is achieved by flying straight, the domain is a distance, in feet, that the vehicle flies before turning toward the target. In cases *III* and *IV* (when the desired path length is found between the initial path and the first critical point), the "x" value of the function is the delay angle, in radians, that the vehicle turns away before turning back towards the target. In every case, the domain will always be positive since the initial vehicle position is the point of zero elongation.

The desired path length will always need to have a window of possible values for the iteration. If an exact value is required the number of iterations to find that path will go to infinity. As a result, a window of possible values will be needed to ensure that a suitable path is found in a reasonable number of iterations. In the present work a window was hard coded into the program at 0.05%. For example, if a path of 15,000 feet is desired, an acceptable path will be between 15,000 and 15,000*1.0005 = 15007.5 feet.

The iteration begins by generating a new path and determining its length given some initial elongation. The initial elongation value for cases *I* and *V* (where elongation is a straight flight) is 55% of the total desired length of elongation. Cases *III* and *IV* have a natural initial elongation equal to a turn to the first critical point. This is because if the path is longer than the path created at the critical point the iterative method must change to a different case. If the path is too long when evaluated for an elongation at the critical point, then it serves as an initial elongation for the iteration.

The iteration begins by linearly connecting the last two computed points in the function. A point is a path length and delay distance pair. The line connecting the points linearly estimates the needed value of the elongation to get the desired path length. The path for the estimated elongation is then created and the new path length is compared to that of the desired length. If the path length is not within the acceptable window, the iteration continues by using the last two points calculated in the iteration. The process is illustrated in Figure 8. In the figure, only three additional paths were computed before the third path was found within the acceptable window of path lengths. When the last two points computed produce an estimated elongation that is infeasible, the two closest points that have path lengths that window the desired length are used instead of the last two computed points. The estimated delay is infeasible when it is estimated to be negative, or when iterating on the initial opposite turn for cases III and IV and the estimated turn delay angle is greater than the first critical point. An infeasible estimated delay can occur due to nonlinear effects in the function.

Simulation Results

The iterative network flow task assignment methodology described above has been implemented in our multi-vehicle, multi-target coordinated-control simulation. The scenario has eight Wide Area Search Munitions performing a search for targets in a rectangular area. The WASM are using a simple "mowing the grass" search pattern. There are up to 5 different target types possible in the simulation, including a "non-target" target type for objects that appear similar to targets but which may be distinguishable as non-targets by the ATR.

For the simulation results presented, eight vehicles are searching an area containing two targets. The targets have an orientation (facing) that has an impact on the ATR process and desired viewing angles, but this will not be discussed as it does not directly affect the task allocation. The search vehicles are initialized in a staggered row formation, with fifteen minutes of flight time remaining, out of a maximum thirty minutes. This assumes that the vehicles have been searching for fifteen minutes and then find a cluster of potential targets.

Figure 9 shows vehicle flight paths and target locations with minimum-length paths. The colored rectangles represent the sensor footprints of the searching vehicles, and the numbers are the target locations. Colored lines show flight paths. Targets are numbered 1,2. As soon as each target is discovered, classification, attack, and possibly verification (if time constraints allow) tasks are assigned for that target. Since the task allocation algorithm is performed each time a task is completed, it is possible for a vehicle's assignment to change based on new target information, although the memory weighting prevents this from happening if a potential new assignment is not a substantial improvement. Both targets are fully prosecuted in this example, although that is not guaranteed, due to potential timing conflicts.

Figure 10 shows vehicle flight paths and target locations with variable-length paths. Whenever the minimum-length path does not satisfy the timing constraints, a new path that satisfies the constraints, and is near the minimum possible path length that satisfies the constraints, is calculated for each vehicle. All of the tasks are again completed, but this time there is less delay in performing each task subsequent task on a target, as the variable- length path planning algorithm tends to generate verification paths that arrive just after the attack is performed (and similarly for attack paths following classification tasks). The critical difference between the two methodologies is that the minimum-length path algorithm can fail to assign a task, as seen in [7]. The variable-length path generation guarantees that all feasible tasks will be completed, if fuel constraints allow.

Conclusions

In this paper we presented a solution to the problem of task allocation for wide area search munitions. The vehicles are capable of searching for targets, performing ATR to classify targets, attack targets, and perform BDA on targets. An iterative application of a network-flow optimization results in efficient decision-making and assignment of tasks to the vehicles. Inclusion of variable-length path planning guarantees that feasible trajectories will always be calculated to assign all required tasks. Simulation results are presented for eight vehicles searching for and attacking two targets within the search area. The network optimization results in an effective allocation of vehicle resources to the required tasks. This method allows assignment of multiple vehicles to a single target, and multiple targets to a single vehicle. The resulting assignment is sub-optimal, but is effective, guarantees that all targets are fully prosecuted, and can be implemented in real-time with relatively low computational requirements.

References

1. Schumacher, C, Chandler, P.R, Rasmussen, S. R., "Task Allocation for Wide Area Search Munitions Via Network Flow Optimization", Proceedings of the 2001 AIAA Guidance, Navigation, and Control Conference.
2. Nygard, K. E., Chandler, P R. , Pachter, M.., "Dynamic Network Flow Optimization Models for Air Vehicle Resource Allocation," Proceedings of the Automatic Control Conference, June 2001.
3. Chandler, Phillip R., Pachter, Meir, ``UAV Cooperative Classification", to appear in Workshop on Cooperative Control and Optimization, Klewer Academic Publishers, 2001.
4. Chandler, Phillip R., Pachter, Meir, "Hierarchical Control of Autonomous Teams," Proceedings of the 2001 AIAA Guidance, Navigation, and Control Conference
5. Ford, L. R Jr. and D. R. Fulkerson, "Flows in Networks," Princeton University Press, Princeton, NJ, 1962
6. Murphy, R.A., "An Approximate Algorithm For a Weapon Target Assignment Stochastic Program," Approximation and Complexity in Numerical Optimization: Continuous and Discrete Problems, editor: P.M. Pardalso
7. Schumacher, C, Chandler, P. R, Rasmussen, S. J., "Task Allocation for Wide Area Search Munitions Via Iterative Network Flow", Proceedings of the 2002 AIAA Guidance, Navigation, and Control Conference.
8. Amarantidis Kostas & Makris Lambros, "Online Lessons on Newton-Raphson Method," http://uranus.ee.auth.gr/lessons/1/, date unknown.

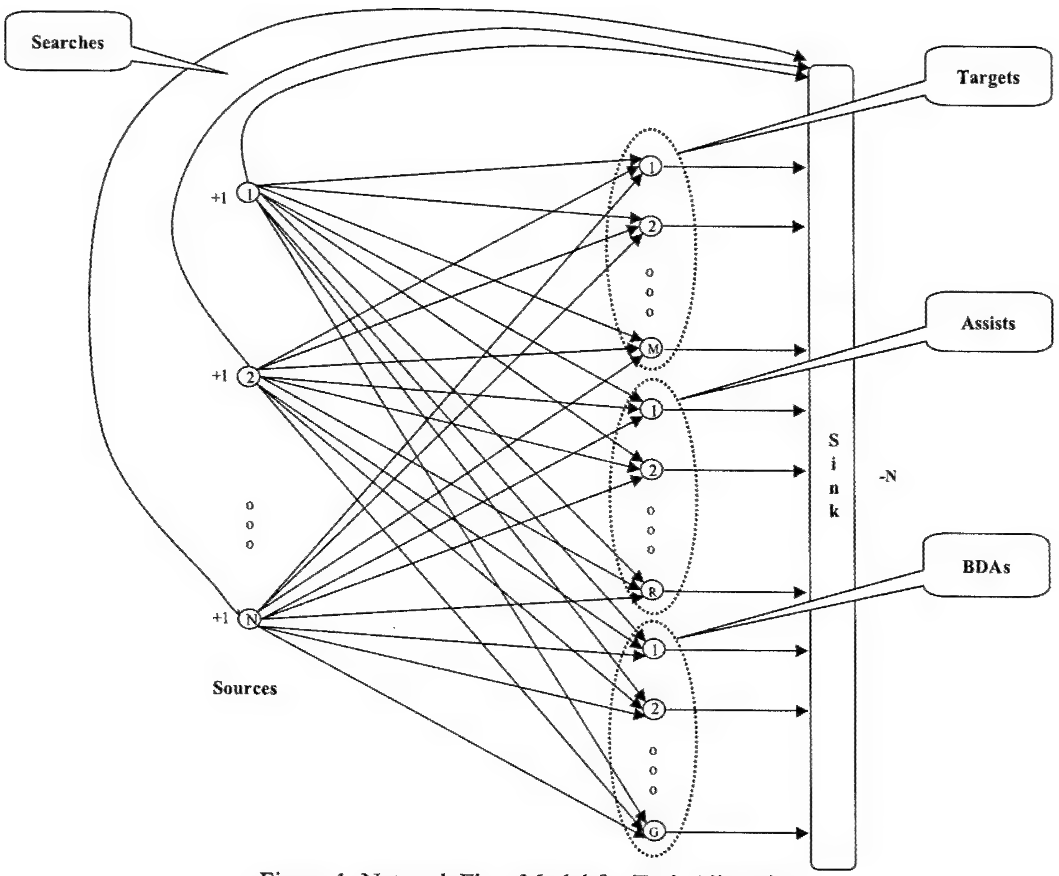

Figure 1: Network Flow Model for Task Allocation

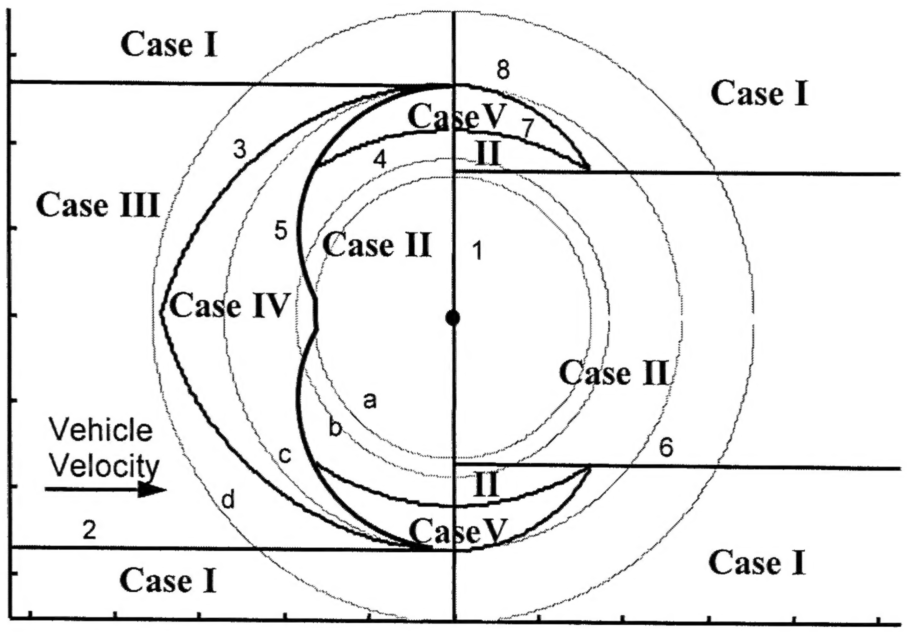

Figure 2 - The Figure shows the boundaries for the various cases. The separate boundaries are numbered in order of use, and are defined in the text. In green, and lettered from a to d are radial distances from the goal critical to the case construction.

Case I

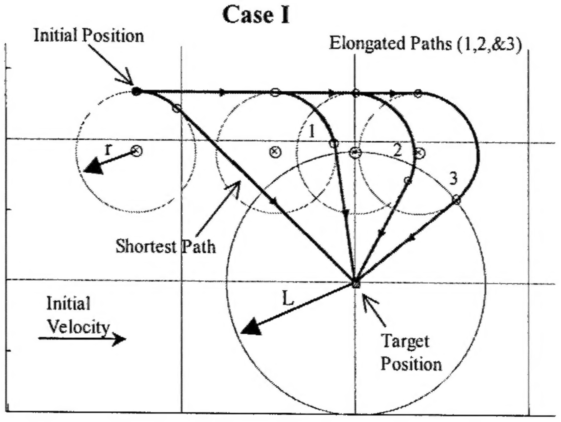

Figure 3 – The trajectory shown on top represents a vehicle in case *I* because it can be elongated indefinitely without path length discontinuities.

Case II

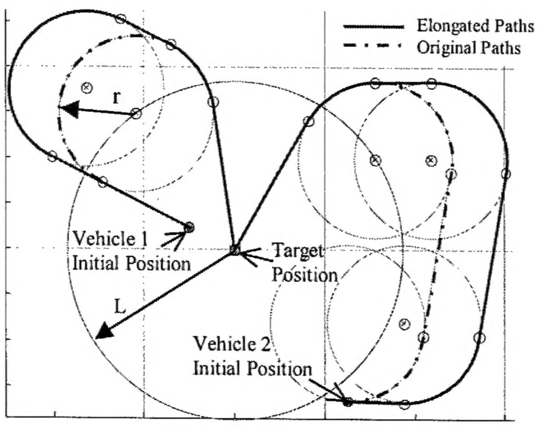

Figure 4 – Shown are two possible paths elongated using the case *II* 180° turn elongation method.

Case III

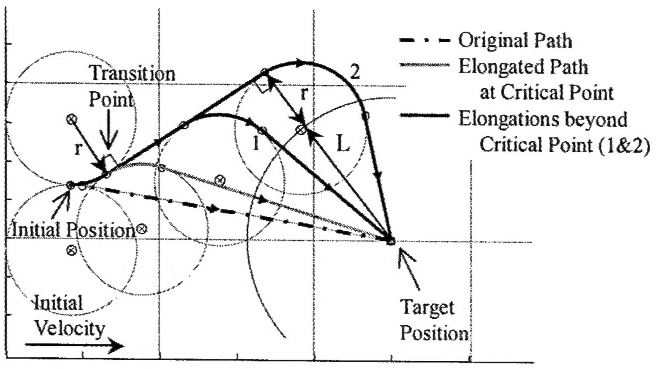

Figure 5 – The figure demonstrates the elongation method associated with case *III*, and shows three elongations. The first elongation is at the transition (or critical) point. The next two elongations are after the vehicle has transitioned to case *I*.

Case IV

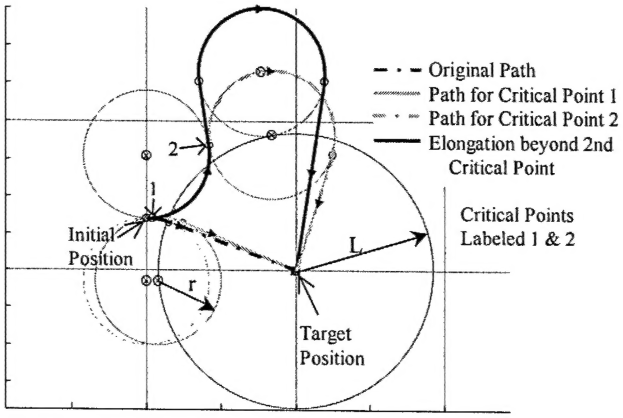
Original Path
Path for Critical Point 1
Path for Critical Point 2
Elongation beyond 2nd Critical Point
Critical Points Labeled 1 & 2
2
1
Initial Position
r
L
Target Position

Figure 6 – The figure illustrates the original path, the paths for elongations to each critical point, and a path beyond the discontinuity for case *IV*.

Case V

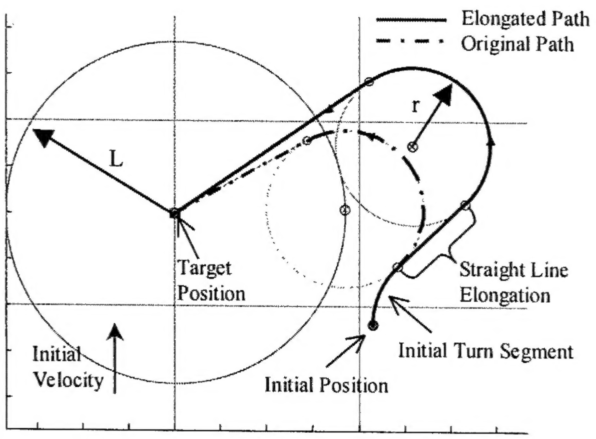

Figure 7 –Elongation for case *V* is performed by switching to case *I* after the initial first turn (which is maintained in the new path).

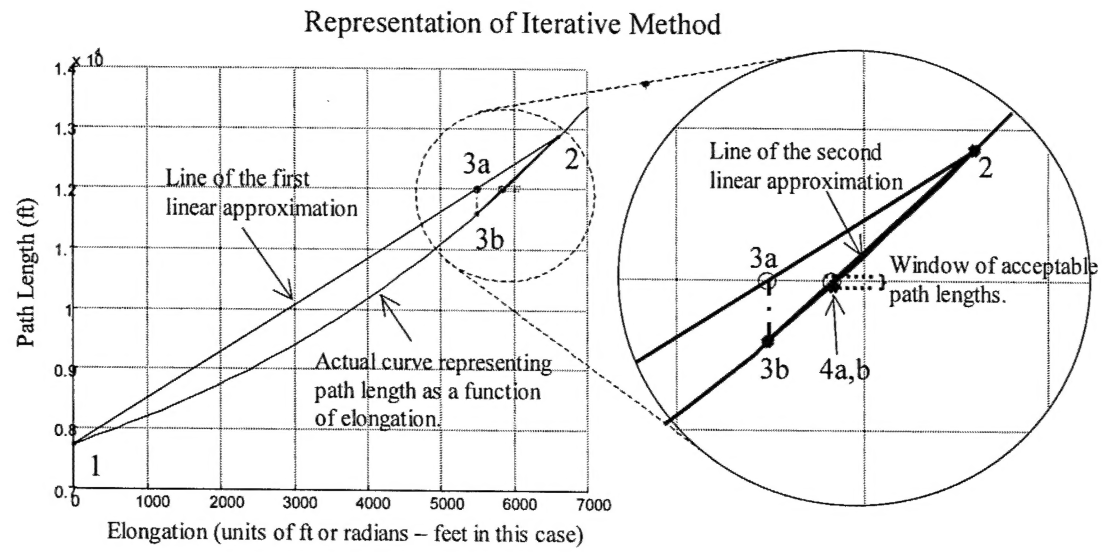

Figure 8 - The figure demonstrates the progress of the iterative method showing the four paths used to find the final path of a desired length. Path 1 is the original, shortest path. Path 2 is the first new path computed. The elongation for paths 3 and 4 are obtained from linear approximations based on previously computed paths. 3a and 4a are the expected path lengths for the elongation. 3b and 4b are the actual path lengths.

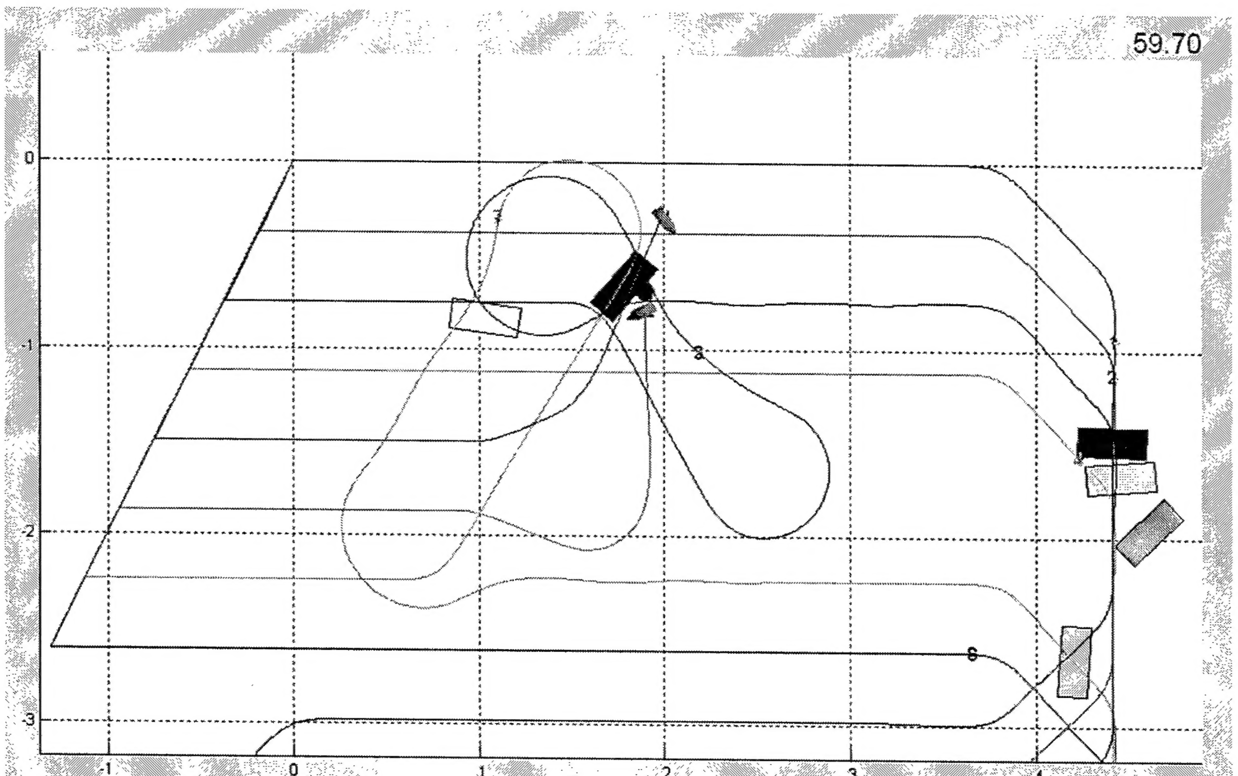

Figure 9 - Vehicle Paths and Target Locations with minimum length paths.

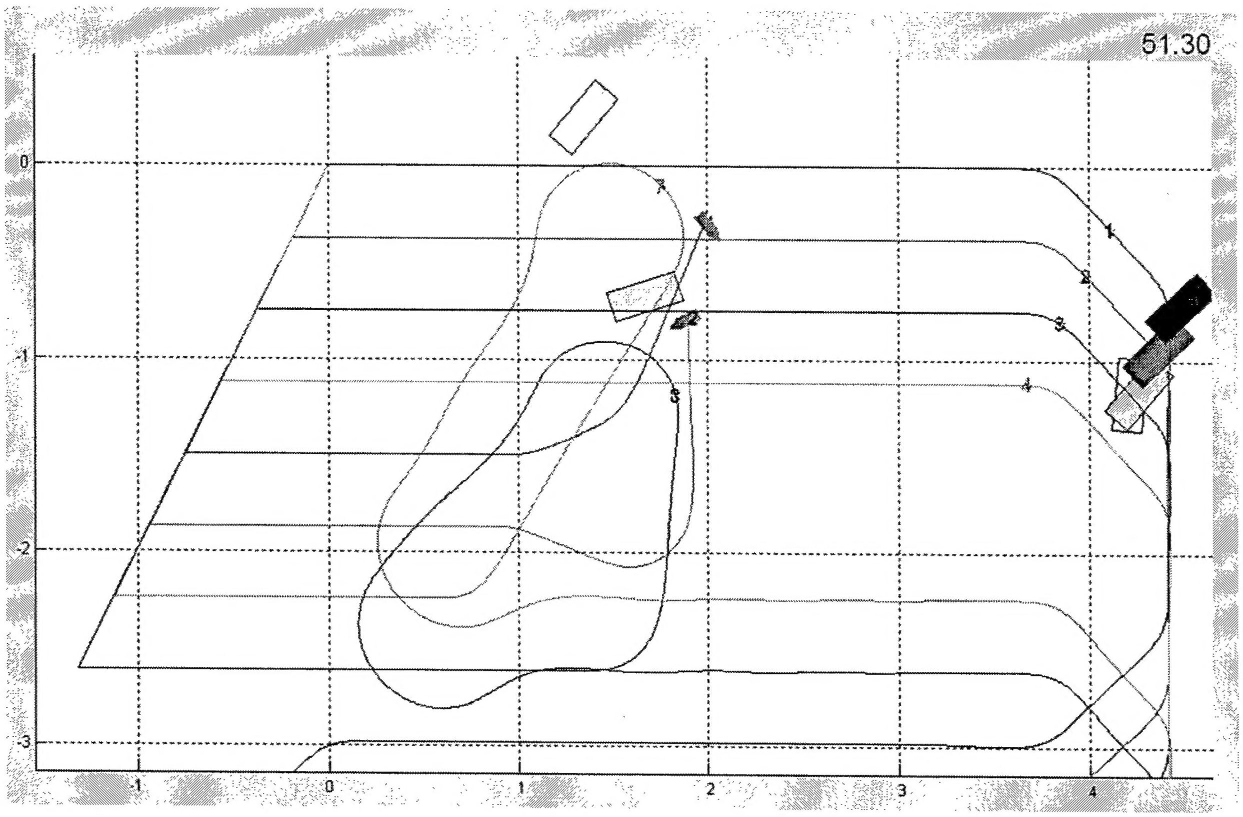

Figure 10 - Vehicle Paths and Target Locations with variable length paths.